HEC FSIO Workshop
HECURA Research Program

August 7, 2008

Walt Ligon
Clemson University

Clemson HEC Filesystem Research

- Two teams
  - Research team
    - ECE Dept
    - HECURA
    - Simulation, metadata, semantics
  - Development team
    - CCIT
    - ACS
    - Server-to-server, caching, security

Project Objectives

- Develop an extensible parallel file system simulation tool
- Study
  - Server-to-server communication
  - Run-time configurable semantics/caching
- Address
  - Scalable metadata
  - Scalable small and unaligned access

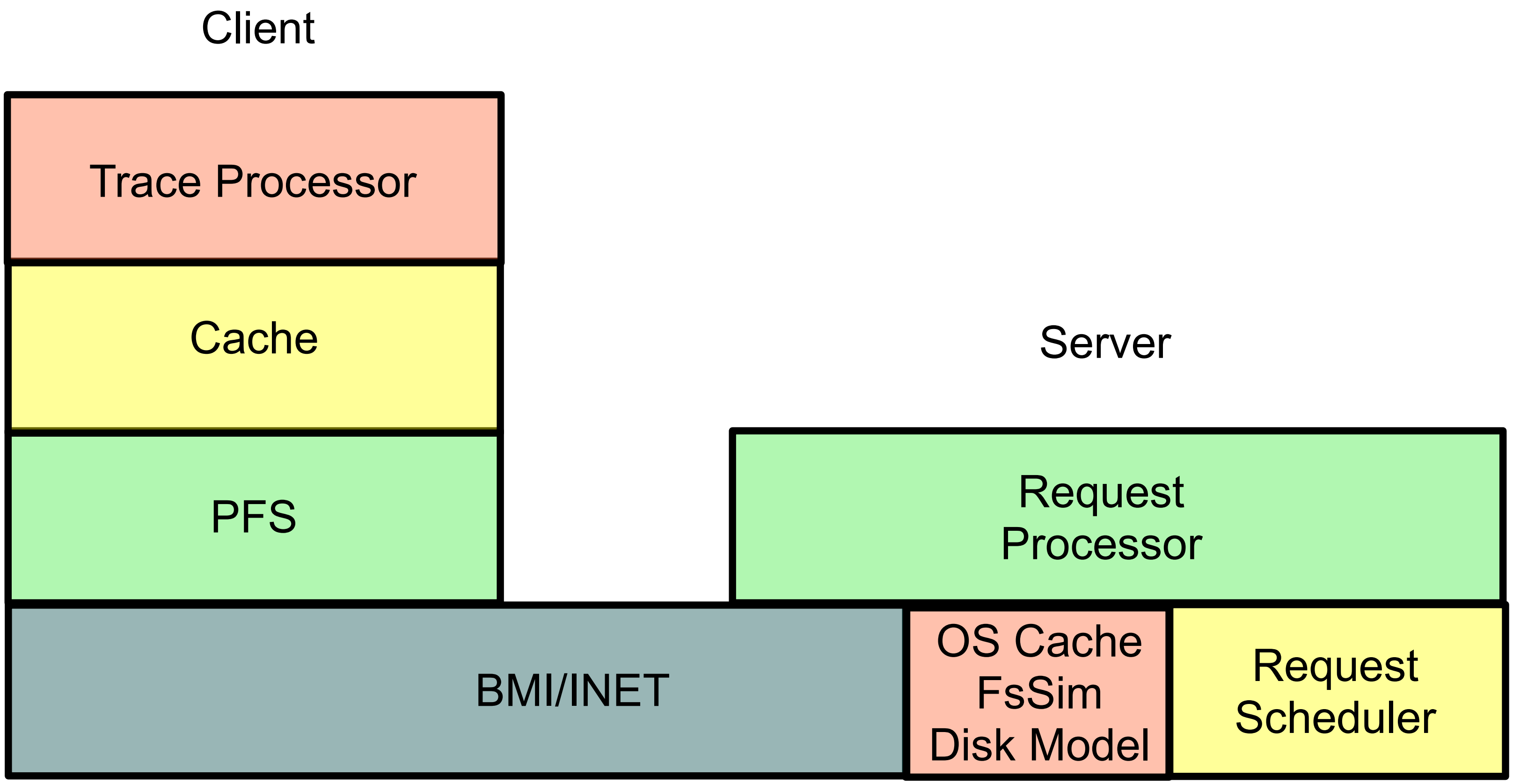
Program Areas Addressed

- Scalable metadata operations
- Scalable small and unaligned operations
- I/O middleware
- Active caching
- Server to server communication
- Simulation of I/O, file, and storage systems

Progress To Date

- Development of HECIOS simulator done
- Tuning and validation
- Scalable metadata
  - Server-to-server communication
  - Collective communication
- Client caching
  - Shared between threads
  - Data layout aware

HECIOS Architecture



Traces

- Developed 2 trace formats
  - SHTF (the serial HECIOS trace format)
  - PHTF (the parallel HECIOS trace format).
  - Both are constructed from ltrace traces
- Successfully used traces from the LANL-trace repository
- Used the LANL Trace library to trace BT-IO and Flash-IO benchmarks

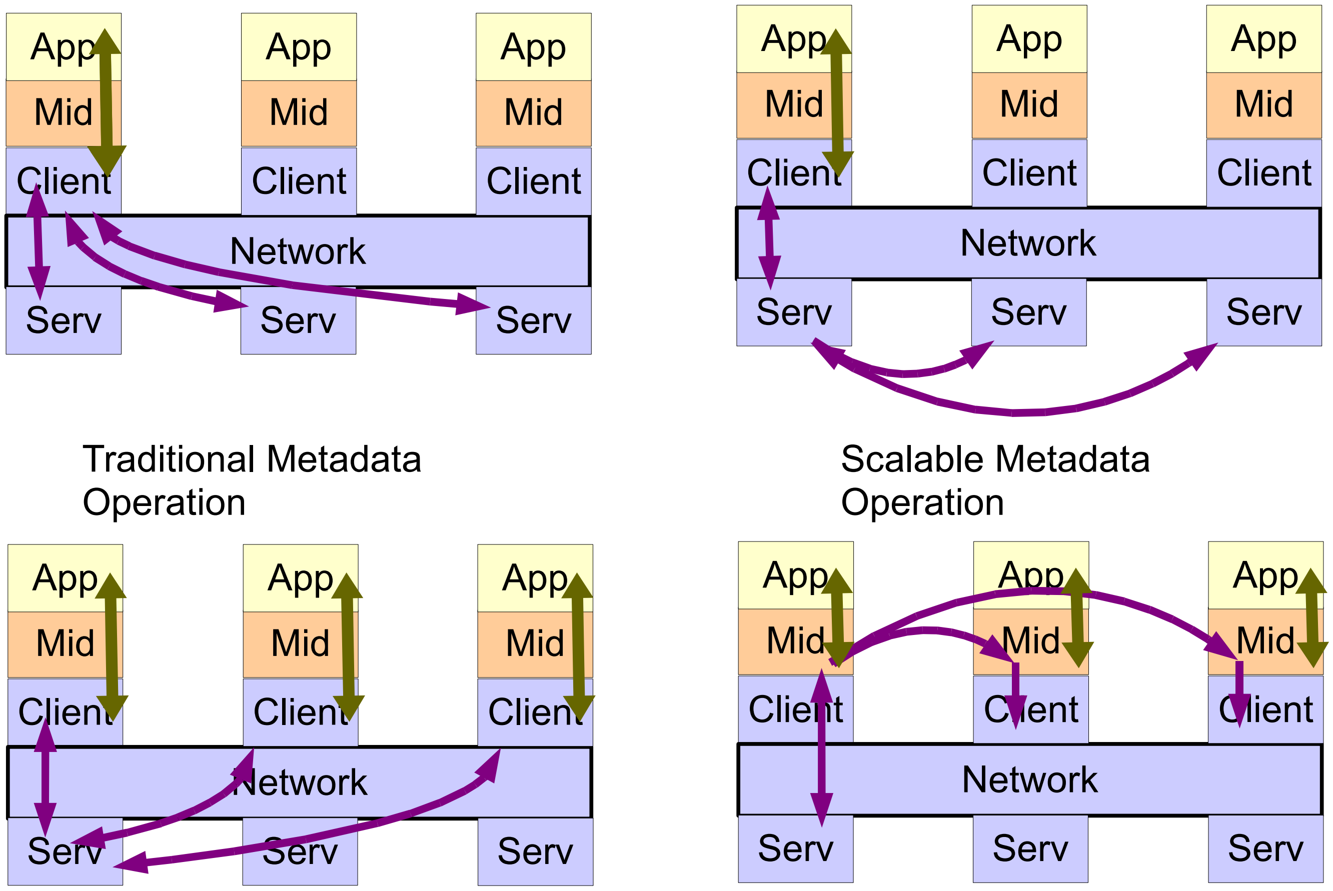
Issues in Trace Library

- Uses ltrace for tracing
  - Modified ltrace regular expressions to capture the fortran MPI calls in BT-IO
- LTrace cannot output more than 5 parameters
  - Created a custom ltrace.conf file to support big MPI calls
- Ltrace won't dereference pointers
  - Wrote a patch for mpich2 that will output those parameters in printf calls
  - Fortran is pass by reference, every call just gives address
  - Ideally, might need to fork ltrace and add this ability

Tuning and Validation

- Tuning
  - Instrumented PVFS
    - Server components (request process, trove, disk time, etc.)
    - Client components (request setup, network overhead, etc.)
- Validation
  - Simple applications – single server (cp, rm, etc.)
  - Phil Carns' prototype results as a comparison
    - Server-to-server/collective communications

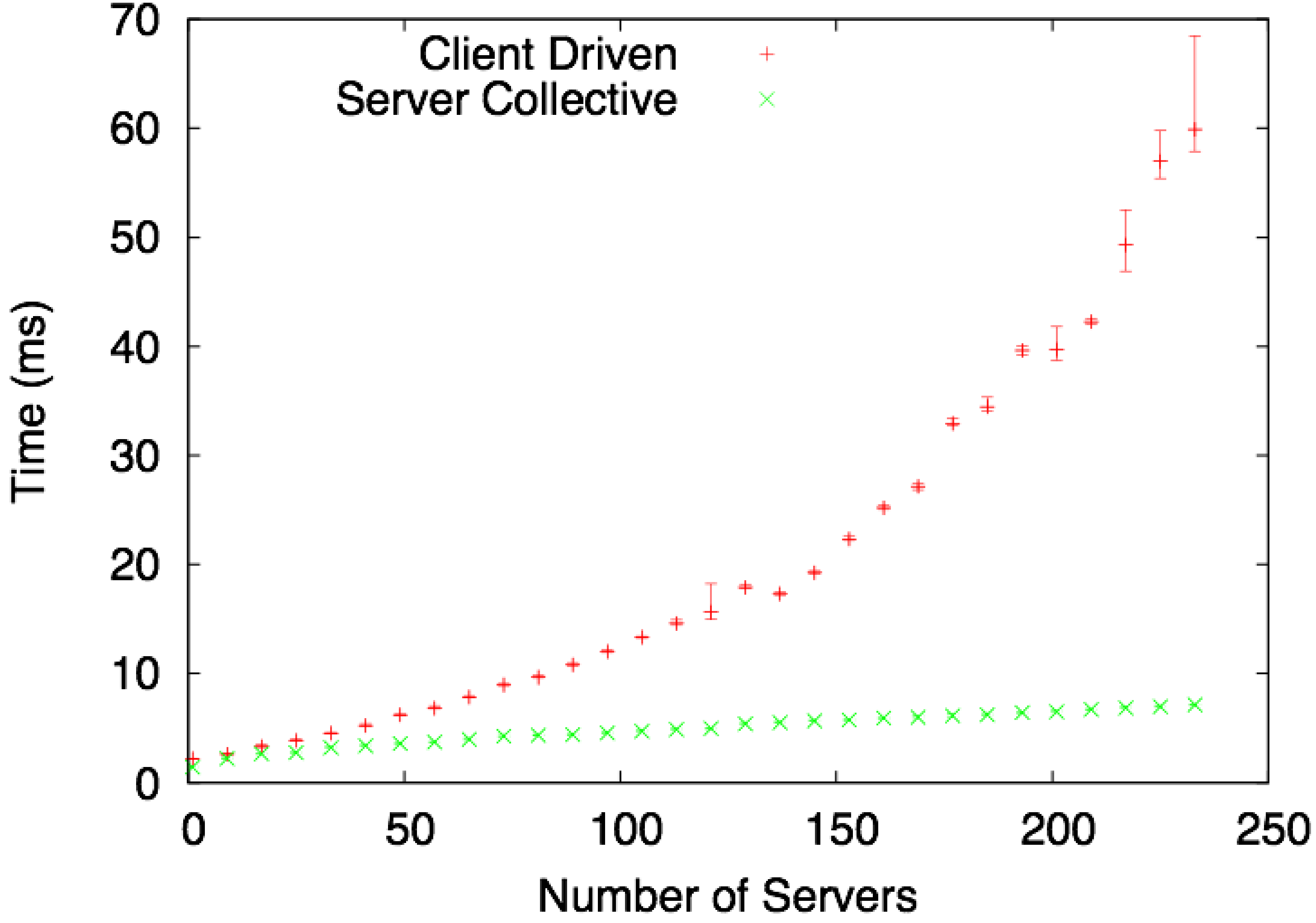
Scalable Metadata
Server-to-Server Communication



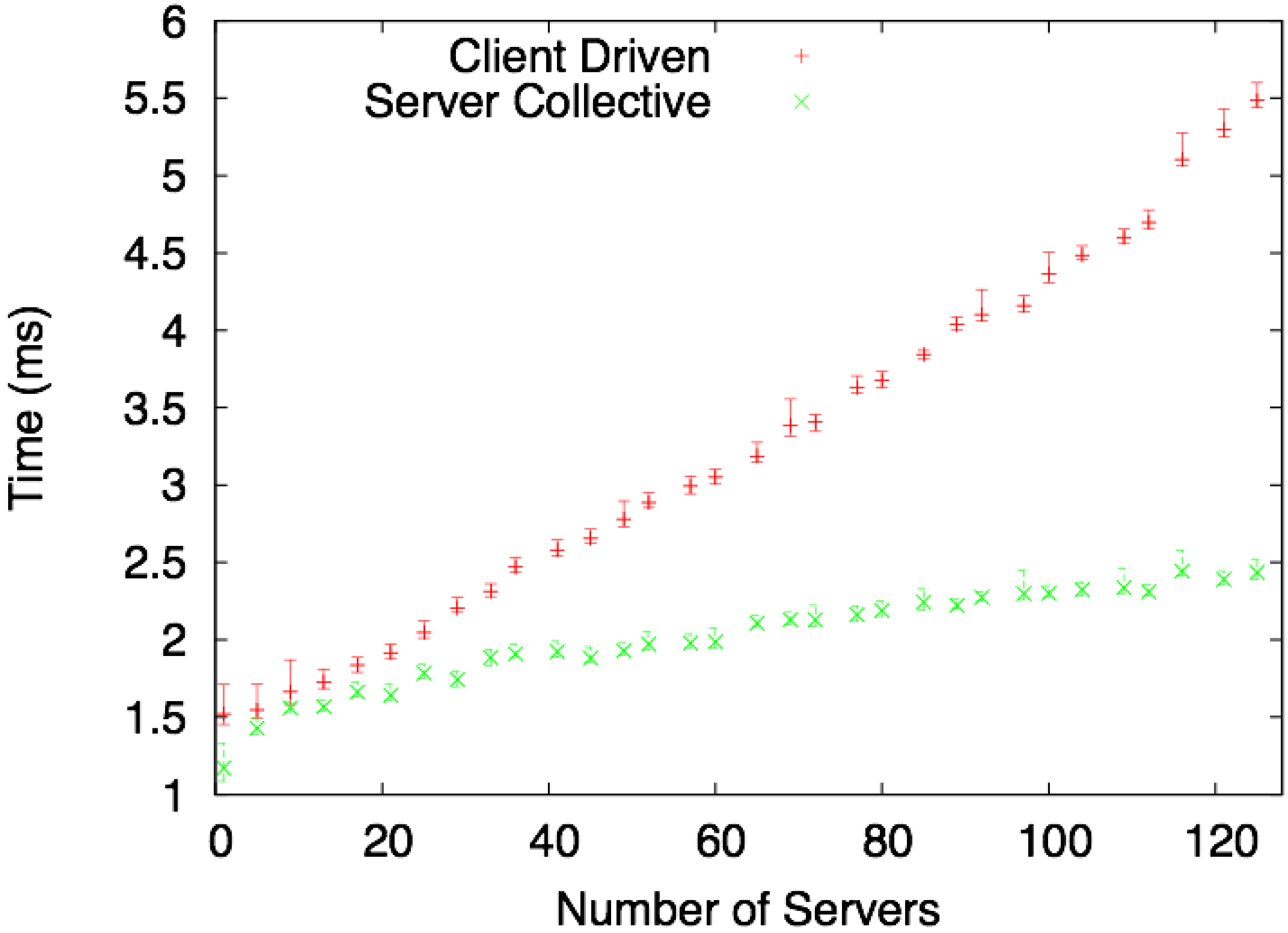
Traditional Metadata Operation

Scalable Metadata Operation

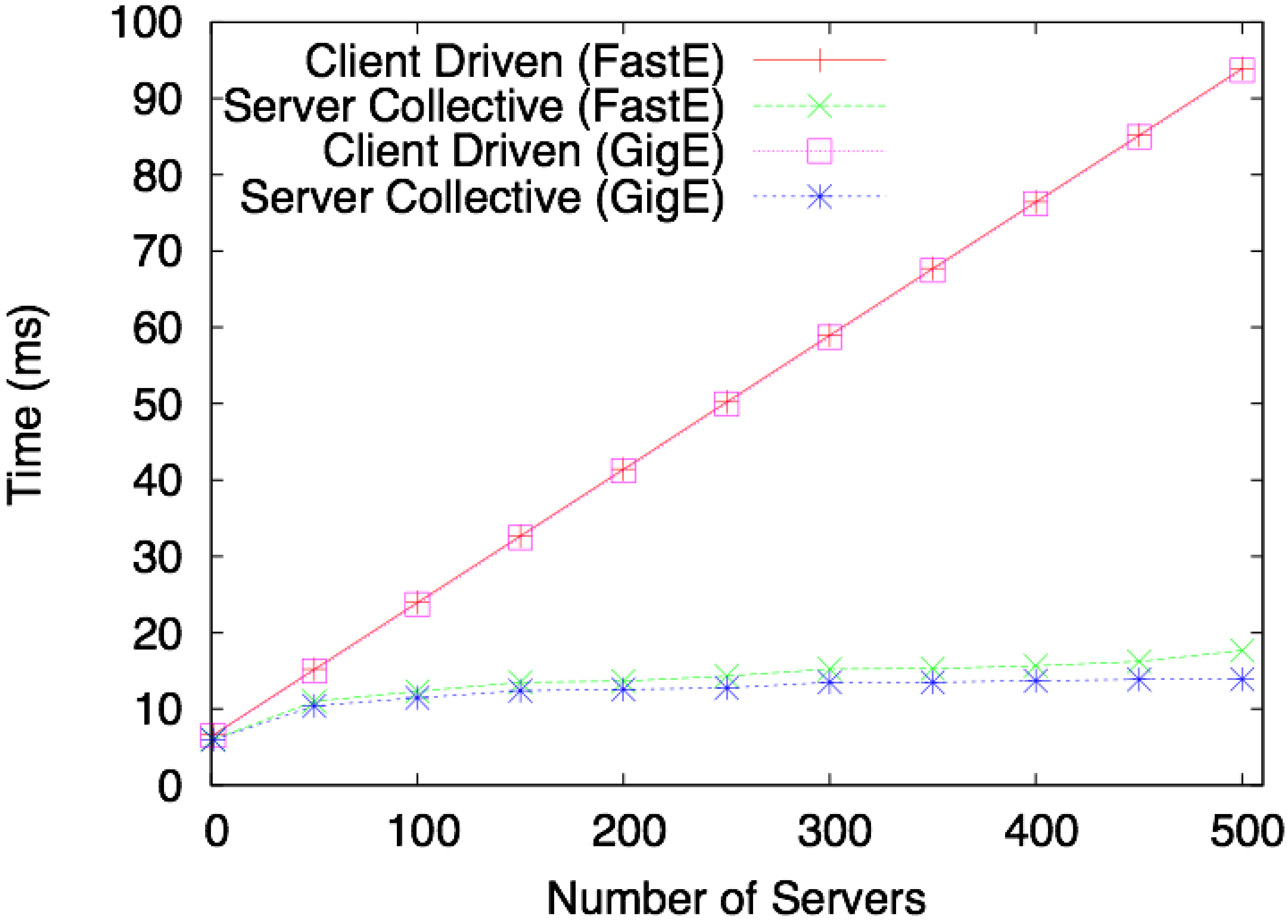
PVFS/TCP Create Time



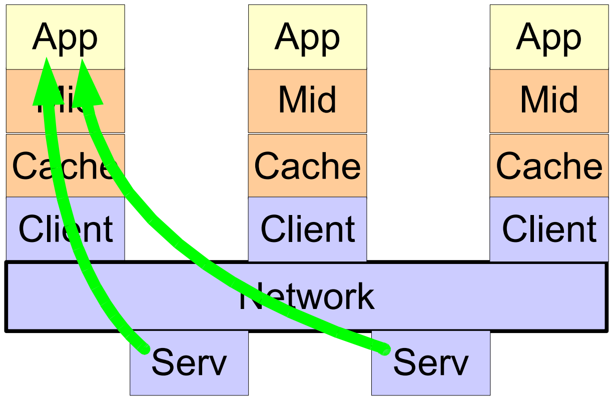
PVFS/GM Create Time



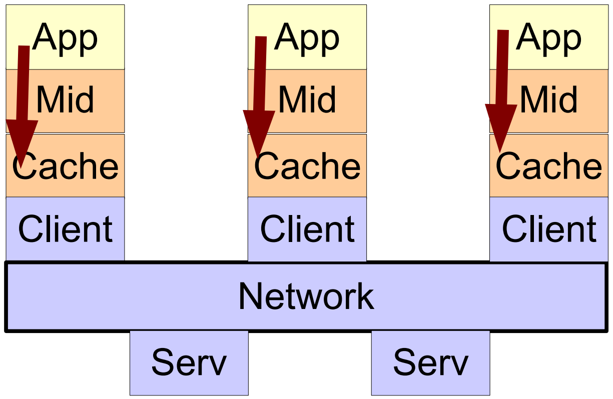
HECIOS Create Times



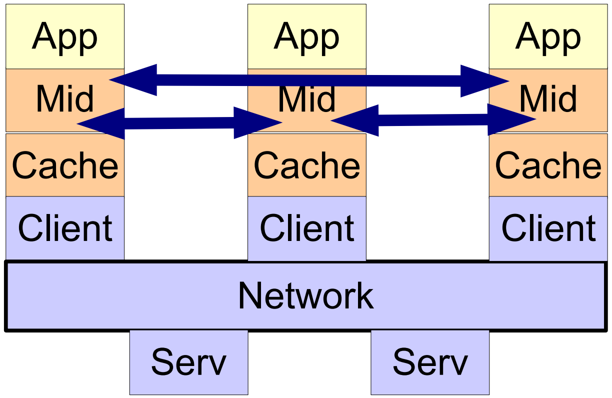
Middleware Managed Cache
Weakened Consistency



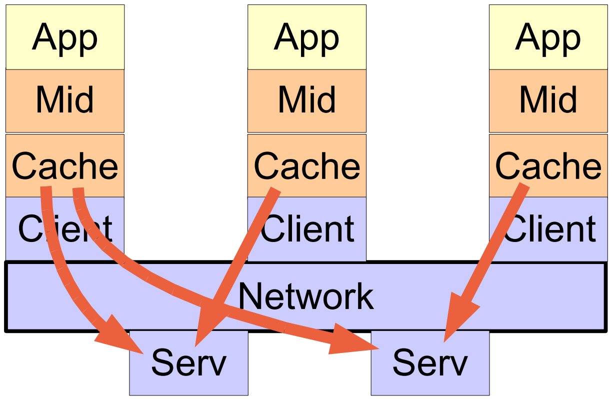
Cold
Read



Write



Synchronization
Event



Write-
back

Middleware Cache Experiments

- Multi-core shared cache
  - cores/cache
  - concurrent access issues
  - size/associativity
- File view based cache
  - FS access efficiency
  - coherency effects
  - combining views

Development Activities

- Server-to-server implementation
  - Metadata operations
  - Redundancy
- Capability-based security
  - External authentication (pam, kerberos, federated)
  - “Unix-level” security
- Middleware caching

New Directions

- The River Model
  - Environment support for building applications
  - Component based
  - Automated memory and IO management
  - Based on DeBardeleben's Coven
  - Modified for script-based applications
  - Brings HEC results back to GP computing